Graduado en Ingeniería Informática

Universidad Politécnica de Madrid

Facultad de Informática

TRABAJO FIN DE GRADO

Generador de exámenes de test aleatorio

Autor: Víctor Navarro Ortego

Director: Juan A. Fernandez del Pozo

MADRID, ENERO DE 2014

Índice

RESUMEN EN ESPAÑOL E INGLÉS

En este documento se detalla, la planificación y elaboración de un paquete que respeta el estándar S4 de programación en lenguaje R. El paquete consiste en una serie de métodos y clases para la generación de exámenes tipos test y soluciones a partir de un archivo xls, que hace las funciones de una base de datos.

El diseño propuesto está orientado a objetos y desarrolla un conjunto de clases que representan los contenidos de una prueba de evaluación tipo test: enunciados, peguntas y respuestas. Se ha realizado una implementación sencilla de un prototipo con las funciones básicas necesarias para generar los tests.

Además se ha generado la documentación necesaria para crear el paquete, esto significa que cada método tiene una página de ayuda, que se podrá consultar desde un terminal con R, dicha documentación incluye ejemplos de ejecución de cada método.

In this document is detailed the elaboration and development of a package that meets the standard S4 of programming language R. This package consists of a group of methods and classes used for the generation of test exams and their solutions starting from a xls format file wich plays the role of a data base at the same time.

These classes have been grouped in a way that the user could have a complete and easy vision of them. This division has been done by using data storage and functions whose tasks are more or less the same.

Furthermore, the necessary documentation to create this package has also been generated, that means that every method has a help page wich can be called from a R terminal if necessary. This documentation has examples of the execution of every method.

INTRODUCCIÓN Y OBJETIVOS

El trabajo consiste en crear una aplicación para generar pruebas tipo test que faciliten el aprendizaje. Se desea diseñar una base de datos de ejercicios de los diferentes temas de los que se pueden extraer preguntas y un programa que explote a demanda dicha base de datos. La aplicación deberá generar el documento final del ejercicio.

El sistema consta de un banco de preguntas; con los datos de dicho banco se deberán aleatorizar las preguntas, las respuestas y los datos (parámetros); además todos los cálculos se deben hacer mediante el lenguaje de programación R y compilar el documento final en un formato del estilo pdf/html/....

La aplicación genera N ejemplares distintos de un mismo examen, las soluciones con las respuestas necesarias de cada uno, la matriz de corrección e incorporar gráficos, tablas y todo el material que se requiera para el examen.

Se ha codificado el TFG mediante el lenguaje de programación R con el paquete Swave, esto se debe a que:

> *R es un conjunto integrado de servicios de software para la manipulación de datos, cálculo y representación gráfica.*
>
> [4]

> *R está diseñado en torno a un lenguaje de programación real, y permite a los usuarios añadir funcionalidad adicional mediante la definición de las nuevas funciones.*
>
> [4]

Por otro lado, para la generación de la versión final se usa LATEX, ya que:

> *"Destinado a la creación de hermosos documentos y especialmente para los documentos que contienen una gran cantidad de formulas matemáticas"*.
>
> [7]

Como sistema operativo, donde se realizará la implementación, se usará Windows, debido a que la posibilidad de usar Excel como Base de datos es muy alta, además de la necesidad de facilitar el uso del paquete a usuarios que no tengan conocimiento en bases de datos. Aunque esto no exime a otros SO, pues el código funciona también en LINUX.

Una vez comentado brevemente que hace y porqué se desarrolla en R, es el momento de explicar el porqué y para qué de este paquete, resulta fácil observar la cantidad de exámenes que se generan al cabo de un corto periodo de tiempo para una sola asignatura, en muchos casos estos exámenes son tipo test, lo que conlleva a no solo pensar la pregunta y la respuesta correcta, sino que también se necesita pensar en respuestas incorrectas.

Cuando se generan muchas preguntas y sus respuestas puede resultar tedioso intentar reutilizarlas, gracias a este paquete, solo se necesitará almacenar las preguntas según se nos ocurran, y generándonos un pequeño programa, usando los métodos del paquete que hemos desarrollado, se conseguirán tantos exámenes distintos como se necesiten.

Sin embargo aún no expliqué por qué es bueno que el software se haga mediante un paquete R, para ello parafrasearé la siguiente cita:

> *Sin lugar a dudas, uno de los grandes puntos fuertes de R es la capacidad de compartir software en forma de paquete. Para el usuario, el paquete proporciona un acceso relativamente fiable, cómodo, y que se ha documentado con una gran variedad de técnicas de código abierto.*
>
> [2]

Por último, me gustaría listar los objetivos que se plantearon al comienzo del proyecto:

- **Definición del problema**: dado un conjunto de cuestiones y las respuestas correspondientes se genera un documento de examen tipo test con un subconjunto de los problemas y cuestiones.

- **Análisis de requisitos**: el orden de los problemas, las preguntas y las respuestas es aleatorio; las respuestas pueden ser literales, gráficas y numéricas; las repuestas pueden ser calculadas automáticamente a partir de parámetros.

- **Codificación de clases y métodos**: se construirá un programa en lenguaje R, conforme a la definición de métodos y clases S4 (John Chambers 2006, http://www.rproject.org/), que satisfaga los requisitos. Se realizará un paquete R que pueda distribuirse e instalarse en el entorno.

- **Documentación y Ejemplos**: el sistema de paquetes R tiene definida la documentación (estructura, contenidos básicos, formatos, referencias,…), por tanto, se realizará la documentación de todas las clases y funciones desarrolladas. En la documentación destacan por su importancia y utilidad los ejemplos, deben ser suficientes para mostrar el uso de las funciones, y el alcance del paquete. Se realizará un banco de preguntas real para su explotación.

TRABAJOS PREVIOS DE LA MEMORIA

Antes de la realización de este trabajo, existían opciones para realizar exámenes tipo test, en su mayoría son programas que generan la estructura de un examen tipo test, pero que obligan al usuario a pensar las preguntas, un ejemplo de estos programas es WebQuestions 2.0.

> *Es un programa gratuito (freeware) creado por Daryl Rowland que de una forma muy sencilla te permite elaborar cuestionarios interactivos en forma de páginas Web sin tener conocimientos de programación.*
>
> [10]

Como este programa existen Moodle y otras plataformas de e-learning que dan facilidades para hacer test, pero son formularios on-line, y no enlazan con un entorno de cálculo.

> *En el lenguaje del módulo Moodle Quiz, que se podría llamar un módulo de prueba. Moodlers pueden referirse a un Quiz como una especie de evaluación de los alumnos. Esta definición puede ser compartida por la palabra "test".*
>
> [9]

Esto sigue sin arreglar el problema de pensar suficientes exámenes para evitar la copia, pues como bien dice la creadora de genertest:

> *Hacer trampa en los exámenes es una práctica común. Los estudiantes que engañan fácilmente a las pruebas, no sólo obtienen aprobados, también son menos propensos a ser motivados en clase. En nuestra experiencia con los estudiantes de medicina, se les dejaba usar sus apuntes y libros durante la prueba, lo que elimina el problema generalizado de chuletas, aún así es muy común atrapar a estudiantes que tratan de copiar unos de otros.*
>
> [6]

Para ello se les ocurrió probar a crear distintos modelos de exámenes, usando las mismas preguntas en cada uno, pero en distinto orden. Esto les hizo descubrir que el porcentaje de alumnos que intentaban copiar disminuía. Gracias a esa observación, pensaron en crear un paquete de R que generara de forma aleatoria exámenes tipo test desde un archivo con las preguntas.

Sin embargo, genertest limita en gran medida como se generan los exámenes y como se construye el banco de preguntas. Por ello, se decidió crear un paquete modular, que de la libertad al usuario de utilizar solo los métodos que necesite, de esta forma se podrán generar exámenes desde un banco de datos propio y teniendo la extensión que el usuario final necesite. Por defecto se dan ya unas opciones para que no se tenga que generar ningún tipo de código en R.

Además para facilitar el uso del paquete, se ha generado documentación y ejemplos sobre los métodos del paquete y el banco de pruebas. De esta forma, cualquier usuario podrá usar el paquete para generar una colección de exámenes tipo test.

DESARROLLO

En el apartado anterior ya se ha expresado la necesidad y el por qué de generar un paquete en R que genere exámenes tipo test. Por tanto, comenzaré explicaré brevemente los pasos que se han seguido, para después pararme en cada uno de ellos, para contarlos detalladamente:

- Análisis de requisitos, se detalla un listado de que necesidades se deben cumplir.
- Planificación, detalle de cuantas horas se asignarán a cada parte del proyecto.
- Diseño del paquete, se generará un diseño de como deberán quedar las clases y su contenido.
- Codificación, siguiendo el diseño anterior se desarrollará el código que lo haga posible.
- Pruebas del paquete, batería de pruebas para verificar el correcto funcionamiento del paquete.

1 Análisis de requisitos

En esta fase del proyecto, se ha leído y estudiado el enunciado del trabajo, de tal forma que se han extraído de él los requisitos necesarios, esto quiere decir, que se han detallado los puntos más importantes y necesarios que deberá cumplir el paquete una vez se hay generado. Por tanto, comienza el listado de requisitos del paquete:

1.1- El paquete tendrá que permitir tener el banco de preguntas de forma sencilla:

1.1.1 Se deberá dar una plantilla de ejemplo y un esquema del banco de preguntas.

1.1.2 El paquete necesitará leer las preguntas de un archivo xls(Tabla Excel).

1.1.3 Se tendrán que devolver errores de lectura en el banco de preguntas, especificando que ocurre.

1.1.4 Se deberá documentar la Base de datos en tabla Excel.

1.1.5 Las preguntas deberán tener atributos tales como: argumentos, temas,… indicando si son o no obligatorios.

1.1.6 Toda pregunta deberá tener una cantidad de respuestas mayor o igual al número de respuestas que se quieran poner en el test.

1.1.7 El banco de preguntas necesitará tener una cantidad de preguntas igual o mayor al número de preguntas que se quieran tener por examen.

1.2- El paquete deberá realizarse en código R, usando el paquete Swave, en colaboración con Latex.

1.2.1 Será necesario documentar con comentarios y ejemplos cada función de las que disponga el paquete.

1.2.2 Se generará cada función por duplicado, una para cada tipo de banco de preguntas.

1.2.3 Se deberá, dar tantos parámetros como sean necesarios, para la generación de test personalizados.

1.3- Se deberán aleatorizar las preguntas, las respuestas y los datos.

1.3.1 Será necesaria una comprobación de los parámetros y respuestas aleatorizadas.

1.3.2 Dentro de lo posible, el paquete deberá poder generar enunciados distintos para una misma solución de problemas.

1.4- Se necesitará que el paquete genere N ejemplares distintos de un mismo examen y las respuestas para cada uno.

1.4.1 Los métodos tendrán que verificar para cada examen, que tenga un número de preguntas suficiente y que todas las preguntas tengan un número necesario de respuestas, de otro se generará un mensaje de error, indicando el problema.

1.4.2 El paquete también deberá pedir como parámetros, cuantas preguntas se quieren, el tema del examen, el número de respuestas,…

1.5- Los exámenes deberán generar un documento pdf.

1.5.1 En caso de error se deberá generar un documento con el error o informar mediante un mensaje.

Estos son los requisitos que se decidieron para este trabajo, sin embargo no se ha explicado por qué se tomo la decisión de usar un archivo xls, o lo que es lo mismo una tabla Excel.

> *Los programadores han producido APIs para abrir hojas de cálculo de Excel en una variedad de aplicaciones y entornos distintos a Microsoft Excel.*
>
> [11]

Esta medida fue tomada pensando en llegar al mayor número de usuarios posibles, casi todas las personas, por no decir todas las personas, que sepan usar R al mínimo nivel, que es el requerido por este paquete, seguramente sepan usar una tabla, por tanto podrían rellenarla con los datos de las preguntas.

Por otro lado, el diseño de la tabla excel es muy simple y gracias a sus columnas podrá rellenarse con gran facilidad, el número de respuestas no se limita, ya que siempre que se ponga "AnswerX", donde X es el número de respuesta, el paquete podrá reconocer la columna como un columna de respuestas. Un ejemplo de cómo es la tabla se podrá apreciar más adelante en el apartado de pruebas.

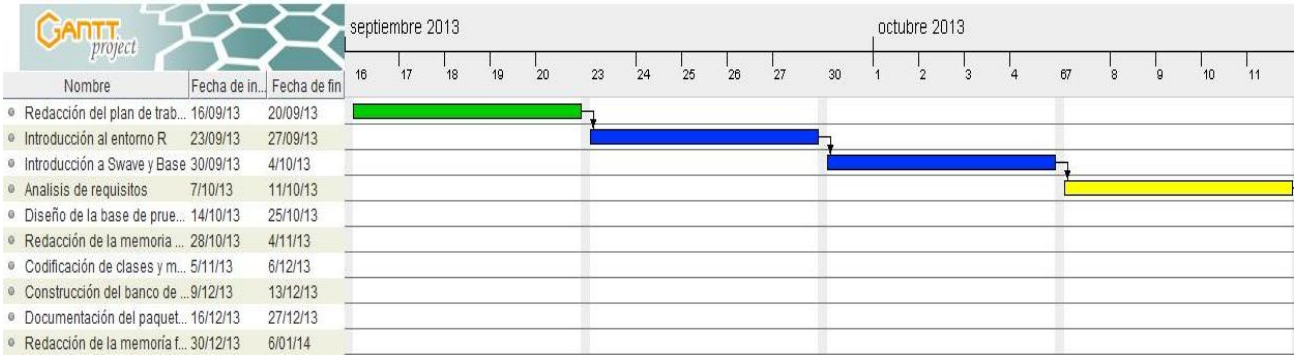
2 Planificación

Una vez realizado el análisis de requisitos podemos planear como abordar el proyecto, de tal forma que podamos asignar tiempos y plazos para cada una de las partes del mismo. Para ello se creó un diagrama de Gantt, con el cual se puede observar que actividad depende de cual, en este caso, todas dependen de la anterior por ser un proyecto lineal.

Antes de explicar el diagrama, comentaré el significado de los colores dentro del diagrama, este significado es relativamente sencillo, indica el aumento de coste de realizar un cambio en esa sección del proyecto:

menor coste → mayor coste

Comenzaré hablando de la primera fase, esta consiste en la familiarización con el lenguaje que se usará y los paquetes, una vez se puede saber de qué utilidades se dispone, es decir, que ventajas y desventajas encontraremos. Tras esto se realiza el análisis de requisitos, esto sería imposible de realizar sin haberse familiarizado con las herramientas.

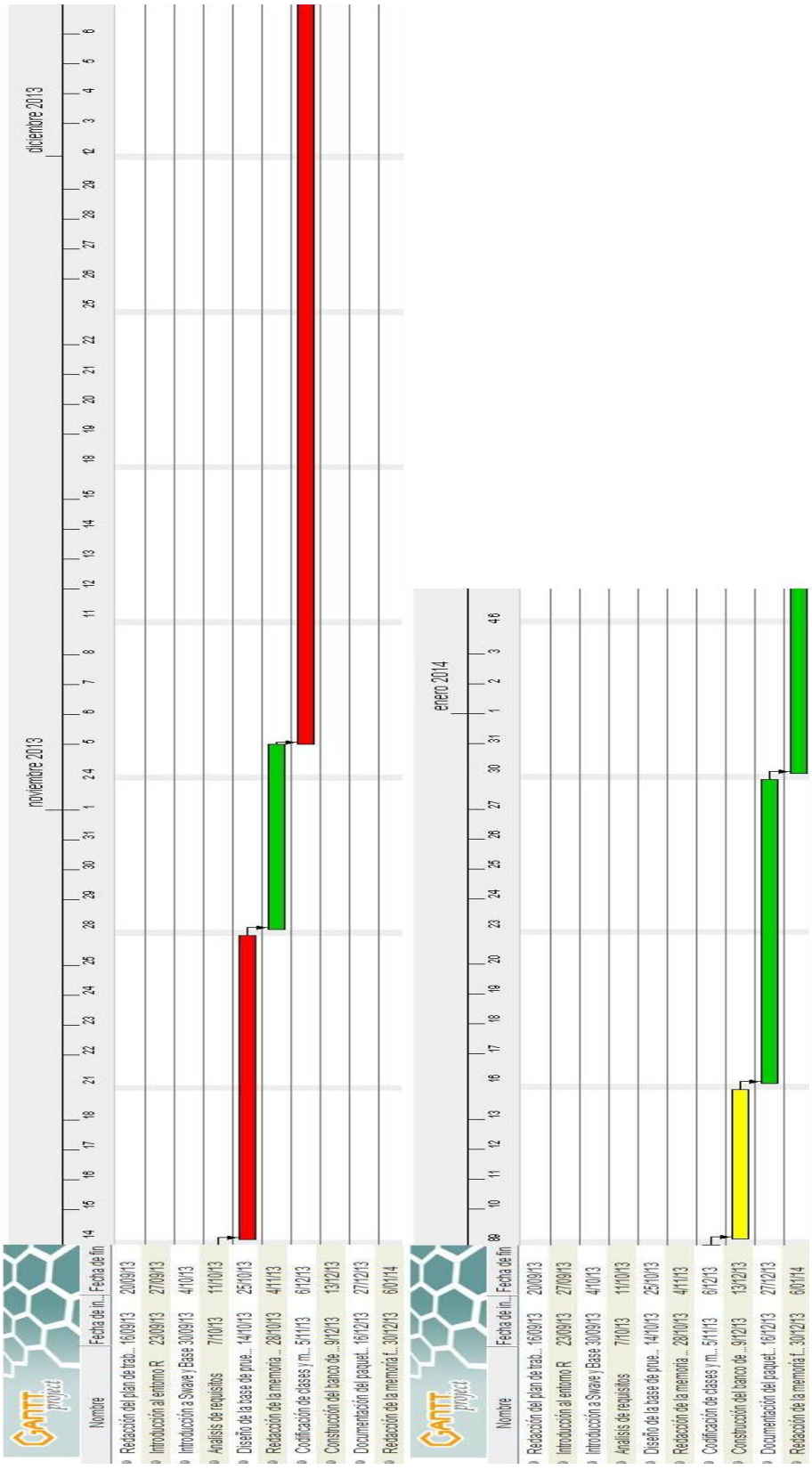


Nombre	Fecha de in...	Fecha de fin
Redacción del plan de trab...	16/09/13	20/09/13
Introducción al entorno R	23/09/13	27/09/13
Introducción a Swave y Base	30/09/13	4/10/13
Analisis de requisitos	7/10/13	11/10/13
Diseño de la base de prue...	14/10/13	25/10/13
Redacción de la memoria ...	28/10/13	4/11/13
Codificación de clases y m...	5/11/13	6/12/13
Construcción del banco de ...	9/12/13	13/12/13
Documentación del paquet...	16/12/13	27/12/13
Redacción de la memoria f...	30/12/13	6/01/14

Esta fase se realiza antes del análisis de requisitos, pues es necesario asignar tiempos a esta parte, para evitar, que se dediquen esfuerzos excesivos a alguna de estas partes.

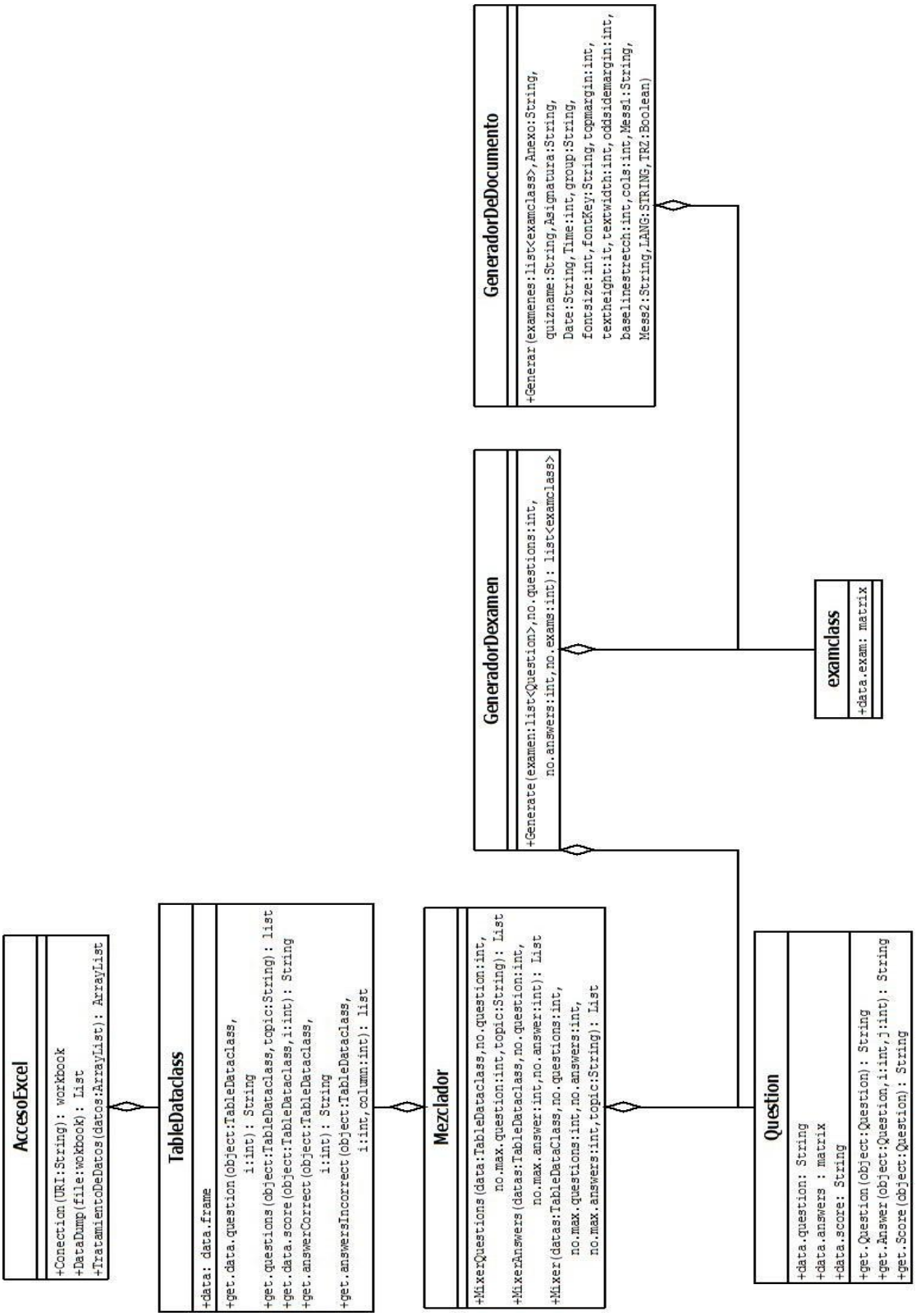
La segunda parte de la planificación, es la parte más crucial, pues en ella se contiene la codificación del proyecto, aunque no solo eso, también la generación de la documentación, es la parte que contiene más tiempo por la siguiente razón:

> *Definición de clases y métodos formales es una carga adicional, en comparación con sólo sumergirse en analizar los datos, o incluso en comparación con la programación mediante la escritura de funciones para encapsular el análisis.*
>
> *[2]*


noviembre 2013
diciembre 2013
enero 2014
Nombre
Fecha de in...
Fecha de fin
Redacción del plan de trab... 16/09/13 20/09/13
Introducción al entorno R 23/09/13 27/09/13
Introducción a Swave y Base 30/09/13 4/10/13
Analisis de requisitos 7/10/13 11/10/13
Diseño de la base de prue... 14/10/13 25/10/13
Redacción de la memoria ... 28/10/13 4/11/13
Codificación de clases y m... 5/11/13 6/12/13
Construcción del banco de ... 9/12/13 13/12/13
Documentación del paquet... 16/12/13 27/12/13
Redacción de la memoria f... 30/12/13 6/01/14

3 Diseño del paquete

En esta parte, se genera un diagrama de clases, este diagrama estará basado en el estándar S4, el cual contiene como han de ser las clases en R, para generar un código basado en Objetos. De esta forma no solo, se hace un diseño que siga un estándar, sino que también se prepara el paquete para poder ser subido al repositorio oficial.



En el diagrama se pueden apreciar clases que corresponderían a objetos, estas son TableDataClass, Question y examclass. Estas clases están pensadas para simplificar el manejo de datos. Examclass podría contener al objeto Question, pero ello requeriría que se limite el uso del paquete. Tal y como se ha diseñado, cualquier persona podría generar una matriz con preguntas y que el Generador de Documentos se encargue de crear los pdf.

Por otro lado esto hace que en caso de aumentar la magnitud del proyecto, se pudiera asignar más gente en el, sin ninguna necesidad de conocer el funcionamiento de otras clases. Esto significa que se simplifican las relaciones entre clases, otorgando una organización jerárquica simple.

> *New Classes*
> *This chapter presents techniques for defining new classes of R objects. The key challenge is dealing with complexity and growth: how to expand the computing capabilities in a way that is easy to use and leads to* trustworthy software.

[2]

Además, con este diseño se logra separar los elementos que generan el examen, de los que contienen los datos, se podrían usar los contenedores de datos sin usar los contenedores de métodos pero no al revés. Esto facilita la comprobación de datos y se evitan errores de una mala generación de elementos.

4 Codificación

En esta fase se creó el paquete, por lo tanto procederé a comentar que hace cada clase con más detalle, que en el apartado de diseño.

TableDataClass

Esta clase genera un objeto que contiene los datos de la tabla Excel, sin embargo no acaba ahí su utilidad, también tiene métodos que facilitan la extracción de los datos. De esta forma solo rellenando una tabla con el formato adecuado se pueden extraer los datos, por tanto una persona se puede encargar de rellenar la tabla y otra sin saber que datos contiene extraer los datos.

AccesoExcel

Esta clase es solo un contenedor de métodos. Podría estar dentro del propio TableDataClass, pero para que resulte más fácil entender el código, se han agrupado los métodos según su tarea. Esta clase tiene los métodos que abren el archivo .xls y lo leen, creando así un objeto de la clase TableDataClass.

Question

En esta ocasión, la clase genera objetos que contendrán los valores de la pregunta, lo que significa que contiene, la pregunta, la respuesta correcta, las respuestas incorrectas y la puntuación asignada a la pregunta. Además dentro de la clase se encuentran métodos que ayudan a la extracción de cada una de las partes del objeto.

Mezclador

Como el propio nombre de la clase ya deja intuir, es la clase contenedora de los métodos para mezclar, tanto preguntas como respuestas. Generando así un examen, en esta ocasión el examen no es de tipo examclass, sino que es una lista de objetos de la clase Question.

Examclass

Para esta clase no se ha complicado mucho el sistema de almacenamiento, sino todo lo contrario, se buscó simplificarlo al máximo, para ello se recurre a una matriz donde se almacenan las preguntas, las respuestas ya desordenadas, la posición de la respuesta correcta y la puntuación. Esta decisión de diseño se tomó, tras pensar en la posibilidad de que el usuario final haga el mezclado de preguntas de forma propia, incluyendo el meter una a una las preguntas.

GeneradorDeExamen

Esta clase recibe una lista de objetos de la clase Question y devuelve una lista con objetos de la clase examclass, por lo tanto solo tiene un método, el cual se encarga de generar tantas copias diferentes del examen como se le pidan, siempre conservando las mismas preguntas y respuestas pero todas ellas en distintos ordenes.

GeneradorDeDocumento

Por último queda comentar la clase que genera los archivos pdf con las soluciones y los exámenes, sin embargo esta clase debería ser la que los usuarios pudieran generar de forma propia, pues aunque se dan muchas opciones de personalización de los exámenes, sigue teniendo un formato rígido, por lo tanto un usuario habilidoso podría crear su propio generador de documentos y crear un sistema de test online donde se crean los exámenes aleatorios pero en lugar de en un documento pdf en un documento php.

Para la creación de esta clase se recibe ayuda de otros sistemas, el primero de ellos es Perl, mediante el paquete latexpand, el cual genera el .tex evaluando la macro \input, se eligió perl por la siguiente razón:

> *Perl fue pensado para que fuera práctico (facilidad de uso, eficiente, completo) en máquinas con poco rendimiento. Sus principales características son que es fácil de usar, soporta tanto la programación estructurada como la programación orientada a objetos y la programación funcional, tiene incorporado un poderoso sistema de procesamiento de texto y una enorme colección de módulos disponibles.*

[14]

Por otro lado quién en realidad genera el archivo pdf es LaTex con Sweave.sty, básicamente interpreta el archivo creado con Perl y lo transforma en un archivo pdf. Como se puede ver Perl procesa textos también y por tanto es perfecto para LaTex:

> *LaTeX es un sistema de composición de textos que está formado mayoritariamente por órdenes construidas a partir de comandos de TeX —un lenguaje «de nivel bajo», en el sentido de que sus acciones últimas son muy elementales— pero con la ventaja añadida de «poder aumentar las capacidades de LaTeX utilizando comandos propios del TeX descritos en The TeXbook».*

[12]

NOTA: EL código de cada una de las clases se encuentra en el Anexo.

5 Pruebas

Para esta fase se ha generado varios bancos de preguntas para verificar el correcto funcionamiento del programa, un vez se verificó que el programa funcionaba correctamente, se procedió a generar el paquete, según se indica en el cran de R, de tal forma que se documento todo el código, creando las páginas de ayuda que se pueden invocar desde R con '?' seguida del método en el cual se tiene la duda de funcionamiento. Cada método, dentro de la página de ayuda, tiene un ejemplo con el que se puede ver la llamada al método correspondiente y que métodos se necesita llamar para su correcto funcionamiento.

El contenido del paquete respeta lo dicho en el manual para generar paquetes:

> *Las fuentes R de un paquete consisten en un subdirectorio que contiene archivos de descripción, un espacio de nombres y los subdirectorios R que contienen: datos, demos, exec, inst, man, po, src, pruebas, herramientas y viñetas (algunos de los cuales puede faltar, pero no deberían estar vacíos). El subdirectorio del paquete también puede contener archivos INDEX, de configuración, de limpieza, licencias y noticias. Otros archivos como Instaladores, README/README.md2 o ChangeLog serán ignorados por R, pero pueden ser útiles para los usuarios finales.*

[18]

Tras generar el paquete correctamente se instalo en un ordenador que no tuviera el código del paquete, de tal forma que nos aseguramos de que sea el paquete quien es llamado. Una vez instalado se repiten las pruebas anteriores y se verifica que las páginas de ayuda sean correctas.

Esto son las pruebas, pero aún no se comentó de forma clara en qué consiste la creación del paquete. El primer paso es el más obvio, consiste en generar el código R que queremos aparezca en el paquete, una vez hecho esto debemos generar el esqueleto de paquete, para ello cargamos con source() todos los archivos que queramos que aparezca, algo a tener muy en cuenta, es que no pueden coexistir métodos con el mismo nombre aún siendo de archivos diferentes, esto es debido a que al generar el paquete, se generaran los archivos de ayuda .Rd, que será los que contengan los archivos para el man.

Una vez cargados los archivos y verificado que no tenemos colisión de nombres, llamamos al método package.skeleton(), el cual genera una carpeta con la estructura del paquete, sin embargo los archivos creados están vacios, por lo que deberemos rellenarlos, siempre teniendo el cuidado de no volver a llamar a package.skeleton() o perderemos el trabajo realizado.

Un ejemplo de estos archivos .Rd es el siguiente:

```
\name{Conection}
\alias{Conection}
%- Also NEED an '\alias' for EACH other topic documented here.
\title{function to connect to a Database}
\description{
  This function is used to create a connection to an XLS file, so that later can be modified or read that file.
}
\usage{
  Conection(URI = "Banco.xls")
```

```
}
%- maybe also 'usage' for other objects documented here.
\arguments{
  \item{URI}{
   URI is the String variable containing the path or filename XLS File
   }
}
\details{
%%  ~~ If necessary, more details than the description above ~~
}
\value{
    \item{workbook}{workbook variable type, which is used to access an XLS file}
}
\references{
%% ~put references to the literature/web site here ~
}
\author{
   Victor Navarro Ortego
}
\note{
   To use this method the XLConnect and rJava libraries are required
}

%% ~Make other sections like Warning with \section{Warning }{....} ~

\seealso{
%% ~~objects to See Also as \code{\link{help}}, ~~~
}
\examples{
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--         or do  help(data=index)  for the standard data sets.
   con<-Conection("Banco.xls")


## The function is currently defined as
function (URI = "Banco.xls")
{
    library("XLConnect")
    archive <- loadWorkbook(URI, create = FALSE)
    return(archive)
  }
}
% Add one or more standard keywords, see file 'KEYWORDS' in the
% R documentation directory.
\keyword{ ~kwd1 }
\keyword{ ~kwd2 }% __ONLY ONE__ keyword per line
```

Para ver el resto de la ayuda se puede acceder al Anexo.

Una vez realizado esto se debe comprobar que no se han cometido errores, ya que una vez generado el paquete si alguien lo usa se encontrará los errores, para ello ejecutamos R CMD check, lo cual nos muestra la siguiente traza:

```
* using log directory ~/home/victor/Escritorio/Codigo/gentest.Rcheck
* using R version 2.14.1 (2011-12-22)
* using platform: i686-pc-linux-gnu (32-bit)
* using session charset: UTF-8
* checking for file ~gentest/DESCRIPTION~ ... OK
* checking extension type ... Package
* this is package "gentest version 1.0"
* checking package namespace information ... OK
* checking package dependencies ... OK
* checking if this is a source package ... OK
* checking if there is a namespace ... OK
* checking for executable files ... OK
* checking whether package "gentest" can be installed ... OK
* checking installed package size ... OK
* checking package directory ... OK
* checking for portable file names ... OK
* checking for sufficient/correct file permissions ... OK
* checking top-level files ... OK
* checking index information ... OK
* checking package subdirectories ... OK
* checking R files for syntax errors ... OK
* checking whether the package can be loaded ... OK
* checking whether the package can be loaded with stated dependencies ... OK
* checking whether the package can be unloaded cleanly ... OK
* checking whether the namespace can be loaded with stated dependencies ... OK
* checking whether the namespace can be unloaded cleanly ... OK
library or require call not declared from: "XLConnect"
package methods is used but not declared
See the information on DESCRIPTION files in the chapter "Creating R
packages of the Writing R Extensions" manual.
* checking S3 generic/method consistency ... OK
* checking replacement functions ... OK
* checking foreign function calls ... OK
* checking R code for possible problems ... NOTE
Conection: no visible global function definition for "loadWorkbook"
DataDump: no visible global function definition for "readWorksheet"
get.answerCorrect: no visible binding for global variable "NQuestion"
get.answersIncorrect: no visible binding for global variable
 "NQuestion"
get.data.question: no visible binding for global variable "NQuestion"
get.data.score: no visible binding for global variable "NQuestion"
get.questions: no visible binding for global variable "Topic"
* checking Rd metadata ... OK
* checking Rd cross-references ... OK
Undocumented code objects:
  "set.ACorrect" "set.Answer" "set.Question" "set.Score"
All user-level objects in a package should have documentation entries.
```

```
See the chapter Writing R documentation files in the Writing R
* checking Rd \usage sections ... OK
* checking Rd contents ... OK
* checking for unstated dependencies in examples ... NOTE
library or require call not declared from: "XLConnect"
* checking examples ... OK
* checking PDF version of manual ... OK
```

Una vez verificado que todo está bien, esto se puede observar, porque no hay ningún mensaje de error ni de warning, podemos proceder a la generación del paquete esto se logra con R CMD INSTALL.

He mencionado ya como se genera, cual es el código y como se hizo todo ello, pero aún no se ha mostrado ningún ejemplo funcional, a continuación se mostrará un pequeño ejemplo tanto de cómo sería la tabla del archivo xls y el resultado de aplicarle el paquete.

La siguiente tabla muestra un pequeño ejemplo de una tabla que contendría preguntas para generar exámenes con gentest:

NQuestion	Question	Correct	Answer1	Answer2	Answer3	Score	Topic
1	¿Quien es la actor principal de la pelicula Armageddon?	Bruce Willis	Billy Bob Thornton	Ben Affleck	Wesley Snipes	2	Ciencia ficción
2	¿Que actriz participo en la pelicula El Hombre sin Sombra?	Elisabeth Sue	Angelina Jolie	Kirsten Dunst	Bridget Fonda	4	Ciencia ficción
3	¿Quien es la actor principal de la pelicula El Hombre Bicentenario?	Robin Williams	Hugh Jackman	Chow Yun-Fat	Tommy Lee Jones	2	Ciencia ficción
4	¿Cual es el nombre de la empresa de Kevin Flynn en Iron man?	ENCOM	X-TRONICS	ARCADIUS	TECGAME	4	Ciencia ficción
5	¿Donde desayuna Audrey Hepburn en Desayuno con diamantes?	En Tiffanys	En casa	En coche	En el Max Caffe	2	Clásico
6	En la pelicula El discurso del rey, ¿Que actor interpreta al Rey Jorge VI?	Colin Firth	Geoffrey Rush	Derek Jacobi	Robert Portal	2	Ciencia ficción
7	En la pelicula El gran Torino, ¿A que grupo asiatico pertenecian los vecinos de Walt Kowalski?	Hmong	Coreanos	Japones	Malayos	4	Drama
8	¿Que pelicula no es de Stanley Kubrick?	El jardin secreto	La naranja metalica	Lolita	Espartaco	2	Drama
9	¿Cuantos prisioneros querian escapar a la vez en La gran evasion?	250	100	50	150	4	Clásico
10	¿Que hay escrito en la pierna escayolada de James Stewart en La ventana indiscreta?	Aqui yacen los huesos rotos de	con amor de	Ya te lo advertí	Acuerdate e tu madre	3	Clásico

Basándonos en la tabla anterior se han generado dos exámenes de 5 preguntas y 3 respuestas, se puede observar como tienen las mismas preguntas y las mismas respuestas pero en distinto orden:

Examen Versión: 1

Número:..... Grupo: .....
... Apellidos:
... Nombre:

Test de Cine ($|n*1$)

——- , Tiempo: 5 minutos

El valor de cada pregunta viene junto al enunciado. La pregunta cuya respuesta seleccionada esté mal resta **1/3**. Es obligatorio entregar el test firmado. Escoger la única respuesta válida marcando claramente a la izquierda con **X**:

//

1. Â¿Cuantos prisioneros querian escapar a la vez en La gran evasion? (4 puntos)
   a) 250
   b) 50
   c) 150
2. Â¿Que hay escrito en la pierna escayolada de James Stewart en La ventana indiscreta? (3 puntos)
   a) Acuerdatede tu madre
   b) con amor de
   c) Aqui yacen los huesos rotos de
3. Â¿Donde desayuna Audrey Hepburn en Desayuno con diamantes? (2 puntos)
   a) En Tiffanys
   b) En un coche
   c) En casa
4. Â¿Cual es el nombre de la empresa de Kevin Flynn en Iron man? (4 puntos)
   a) TECGAME
   b) ARCADIUS
   c) ENCOM
5. Â¿Que actriz participo en la pelicula El Hombre sin Sombra (Hollow man)? (4 puntos)
   a) Angelina Jolie
   b) Elisabeth Sue
   c) Kirsten Dunst

Examen Versión: 1
Test de Cine

——- . Respuestas Correctas, Grupo: ..... . Opción correcta (a)

//

1. Â¿Cuantos prisioneros querian escapar a la vez en La gran evasion? ((4 puntos)
   a) (V) 250
   b) (X) 50
   c) (X) 150
2. Â¿Que hay escrito en la pierna escayolada de James Stewart en La ventana indiscreta? ((3 puntos)
   a) (X) Acuerdatede tu madre
   b) (X) con amor de
   c) (V) Aqui yacen los huesos rotos de
3. Â¿Donde desayuna Audrey Hepburn en Desayuno con diamantes? ((2 puntos)
   a) (V) En Tiffanys
   b) (X) En un coche
   c) (X) En casa
4. Â¿Cual es el nombre de la empresa de Kevin Flynn en Iron man? ((4 puntos)
   a) (X) TECGAME
   b) (X) ARCADIUS
   c) (V) ENCOM
5. Â¿Que actriz participo en la pelicula El Hombre sin Sombra (Hollow man)? ((4 puntos)
   a) (X) Angelina Jolie
   b) (V) Elisabeth Sue
   c) (X) Kirsten Dunst

Examen Versión: 2

Número:..... Grupo: .....
... Apellidos:
... Nombre:

Test de Cine ($|n*2$)

——- , Tiempo: 5 minutos

El valor de cada pregunta viene junto al enunciado. La pregunta cuya respuesta seleccionada esté mal resta **1/3**. Es obligatorio entregar el test firmado. Escoger la única respuesta válida marcando claramente a la izquierda con **X**:

//

1. Â¿Que hay escrito en la pierna escayolada de James Stewart en La ventana indiscreta? (3 puntos)
   a) Aqui yacen los huesos rotos de
   b) con amor de
   c) Acuerdatede tu madre
2. Â¿Cual es el nombre de la empresa de Kevin Flynn en Iron man? (4 puntos)
   a) TECGAME
   b) ENCOM
   c) ARCADIUS
3. Â¿Que actriz participo en la pelicula El Hombre sin Sombra (Hollow man)? (4 puntos)
   a) Angelina Jolie
   b) Kirsten Dunst
   c) Elisabeth Sue
4. Â¿Cuantos prisioneros querian escapar a la vez en La gran evasion? (4 puntos)
   a) 50
   b) 150
   c) 250
5. Â¿Donde desayuna Audrey Hepburn en Desayuno con diamantes? (2 puntos)
   a) En casa
   b) En un coche
   c) En Tiffanys

Examen Versión: 2
Test de Cine

——- . Respuestas Correctas, Grupo: ..... . Opción correcta (a)

//

1. Â¿Que hay escrito en la pierna escayolada de James Stewart en La ventana indiscreta? ((3 puntos)
   a) (V) Aqui yacen los huesos rotos de
   b) (X) con amor de
   c) (X) Acuerdatede tu madre
2. Â¿Cual es el nombre de la empresa de Kevin Flynn en Iron man? ((4 puntos)
   a) (X) TECGAME
   b) (V) ENCOM
   c) (X) ARCADIUS
3. Â¿Que actriz participo en la pelicula El Hombre sin Sombra (Hollow man)? ((4 puntos)
   a) (X) Angelina Jolie
   b) (X) Kirsten Dunst
   c) (V) Elisabeth Sue
4. Â¿Cuantos prisioneros querian escapar a la vez en La gran evasion? ((4 puntos)
   a) (X) 50
   b) (X) 150
   c) (V) 250
5. Â¿Donde desayuna Audrey Hepburn en Desayuno con diamantes? ((2 puntos)
   a) (X) En casa
   b) (X) En un coche
   c) (V) En Tiffanys

Como se puede apreciar cada versión del examen tiene su propia corrección, esto se ha hecho para simplificar la resolución de exámenes, no se necesita estar buscando continuamente la respuesta, si no que solo haría falta comprobar los números de preguntas y su respuesta.

RESULTADOS Y CONCLUSIONES

A lo largo de este documento se ha podido comprobar cómo se han cumplido las metas planteadas en un principio. En las primeras etapas, mientras me familiarizaba con el lenguaje de programación R, creí que la codificación sería más fácil de lo que terminó resultando. Esto es debido, a que la sintaxis es bastante fácil, sin embargo en el momento que se quiere explotar el uso de R, con respecto a tipos de datos, como si fuera un lenguaje orientado a objetos, nos encontramos que se complica, puesto que R no fue diseñado en un principio para usarse de esta forma.

Sin embargo el lenguaje R, no solo tiene limitaciones, si no que cuenta con sus propias ventajas, como pueden ser el tratamiento de matrices y la posibilidad de usar un gran sin número de paquetes, que se encuentran en el repositorio oficial.

Solventado el problema de los tipos de datos, la tarea de documentación fue una buena forma de familiarizarse con estándares de este tipo, pues no es algo que durante la carrera se pueda experimenta, pero si es algo que en un futuro pueda ser de gran utilidad. Ya que todo gran proyecto debe ser documentado, para su después explotación.

Ya que mencioné que se usaron estándares de documentación, también es oportuno comentar el uso del estándar S4 para la codificación de R como si fuera orientado a objetos. Esto es relativamente fácil, si se ha experimentado previamente con lenguajes como java, aunque en ocasiones haya cosas que pueden resultar confusas, como es el caso de los métodos genéricos.

Una vez comentada la experiencia de creación del documento, procederé a comentar posibles mejoras que podrían añadirse al proyecto, las cuales no deberían ser muy complicadas de integrar, puesto que se hizo una separación de clases, que generan una suficiente independencia entre ellas.

Para empezar, una posibilidad bastante obvia es la de añadir la posibilidad de tener las preguntas almacenadas como base de datos, preferiblemente sql, por ser la más común. Aunque no se deberían de descartar otros procedimientos con los que capturar los datos.

Pero no solo se puede mejorar la entrada de preguntas, su tipo y contenido también. Se podrían añadir expresiones evaluables, de tal forma que se podrían poner problemas, incluso se podría hacer que los enunciados pudieran recibir parámetros para rellenar, generando de esa forma enunciados de forma aleatoria. Pero no solo existen mejoras tan complejas, otra posibilidad sería la de permitir añadir imágenes a la pregunta.

Como se puede suponer las mejores con respecto al contenido de las preguntas son muchas, pues existen muchos tipos de preguntas y aún más posibilidades de personalizarlas. Por tanto pasaré a las mejoras de la última etapa, se podrían añadir mejoras en el formato de salida del pdf, pero también se podrían exportar otros tipos, como podrían ser archivos php, que devolvieran un formulario tipo test que se autocorrigiera.

Por último, estimo que se habrá cumplido satisfactoriamente el 90% de las metas planteadas, faltaría mejorar algunas cosas como el banco de preguntas y la personalización de los exámenes. Ya que quedaron demasiado básicas, cuando podrían explotarse hasta sacar cosas muy vistosas o complicadas.

ANEXOS

1 Código

TableDataClass

```
### TableData

setClass ("tableDataclass", contains="data.frame",
    representation (data = "data.frame"),
    prototype (data = data.frame())
    )

#Constructor

tableData <- function( data) {
    new( "tableDataclass", data=data)
    }

### Definicion de metodos
get.data.question<-function (object, i){
    subset(object@data, subset=(NQuestion==i), select=c("Question"))[[1]][[1]]
    }

get.questions <- function (object, topic){
    subset(object@data, subset=(Topic==topic), select=c("NQuestion"))
    }

get.data.score <- function (object, i){
    subset(object@data, subset=(NQuestion==i), select=c("Score"))[[1]][[1]]
    }

 get.answerCorrect <- function (object, i){
    subset(object@data, subset=(NQuestion==i), select=c("Correct"))[[1]]
    }

get.answersIncorrect<-function (object, i, column){
    subset(object@data, subset=(NQuestion==i), select=c(column))[[1]]
    }

 ### TableData
```

AccesoExcel

```
### AccesoExcel

Conection <- function( URI = "Banco.xls") { ## SP, EN
  ## database p
  library("XLConnect")
  archive <- loadWorkbook(URI, create=FALSE)
  return( archive)
}

DataDump <- function(file) { ## SP, EN
  ## database p
  library("XLConnect")
  if(class(file)!="workbook")
    print ("You need a valid Workbook.")
  dat<-tableData(readWorksheet(file, "banco"))
}

### AccesoExcel
```

Question

```
### Question

setClass ("questionclass",
    representation (data.question = "character", data.answers = "matrix",
data.score="character"),
    prototype (data.question="",data.answers = matrix(c(""),nrow = 1, ncol = 1),
data.score ="")
)

#Constructor

question <- function(quest, answers, score) {
    new("questionclass", data.question = quest, data.answers = answers,
data.score=score)
}

#Metodos
    get.Question<-function (object){
        object@data.question
    }

    get.Answer<-function (object,i,j){
        object@data.answers[i,j]
    }

    get.Score<-function (object){
        object@data.score
    }

### Question
```

Mezclador

```
### Mixer

MixerQuestions <- function(datas, no.questions, no.max. questions, topic="") { ## SP, EN
  ## database p
  library("XLConnect")
  if(no.questions<1)
    print ("It takes a number of positive question.")
  else if(no.questions>no.max.questions)
    print ("The order number questions is greater than the number of existing questions.")
  else if(class(datas)!="tableDataclass")
    print (cat("It requires that the data variable is the type tableData and the type object is ",class(datas)))
  else{
    if(topic!=""){
      tem=get.questions(datas,topic)
      te<-list()
      for(o in  1:length(tem))
        te[[o]]=tem[[1]]
      if(length(te[[1]])<no.questions)
        print("Not enough questions in the question bank.")
      pre <- matrix(sample(te[[1]], no.questions, replace=F),nrow = no.max.questions, ncol = 1)
    }
    else
      pre <- matrix(sample(1:no.max.questions, no.questions, replace=F),nrow = no.questions, ncol = 1)
    question = get.data.question(datas,pre[[1]])
    num=pre[1,1]
    punt=get.data.score(datas,pre[[1]])
    questions <- matrix(data=c(""),nrow = no.questions, ncol = 3)
      questions[1,1]= question
      questions[1,2]= num
      questions[1,3]= punt
```

```
        for(j in 2:no.questions){
          question = get.data.question(datas,pre[[j]])
          num=pre[j,1]
          punt=get.data.score(datas,pre[[j]])
          questions[j,1]= question
          questions[j,2]= num
          questions[j,3]= punt
        }
        questions
      }
    }
    MixerAnswers <- function(datas,no.question,no.max.answers,no.answers) { ## SP, EN
      correc=0
      ## database p
      library("XLConnect")
      if(no.question<1)
        print ("It takes a number of positive question.")
      else if(no.answers<1)
        print ("You must enter a valid number of questions.")
      else if(no.answers>no.max.answers)
        print ("The order number of responses is greater than the number of available responses.")
      else if(class(datas)!="tableDataclass")
        print ("It requires that the variable datas is the TableData type.")
      else{
        proba<-100/(no.answers+1)
        res <- sample(1:no.max.answers, no.answers, replace=F)
        answers<- matrix(c(""),nrow = (no.answers+1), ncol = 1)
        respu=get.answerCorrect(datas,no.question)
        answers[1,1]= respu
        for(j in 1:no.answers){
          columna=paste("Answer",res[[j]], sep="")
         respu=get.answersIncorrect(datas,no.question,columna)
          answers[j+1]= respu
```

```
        }
        answers
      }
    }

    Mixer <- function(datas, no.questions, no.max.questions, no.answers,
no.max.answers, topic="") { ## SP, EN
      questions = MixerQuestions(datas,no.questions,no.max.questions,topic)
      questionaux<- new ("questionclass")
      examen<-c(questionaux)
      respu<-matrix(c(""),nrow = no.answers+1, ncol = 1)
      for(j in 1:no.questions){
        answers=MixerAnswers(datas,questions[j,2],no.max.answers,no.answers)
        respu[1,1]=answers[1]
        for(r in 2:(no.answers+1)){
          respu[r,1]=answers[r]
        }
        questionaux<-question(quest=questions[j,1], answers=respu,
score=questions[j,3])
          examen[[j]]=questionaux
      }
      examen
    }


    ### Mixer
```

Examclass

```
### Exam

setClass ("examclass", contains="matrix",
   representation (data.exam = "matrix"),
   prototype (data.exam = matrix())
)

#Constructor
exam <- function(exam) {
   new( "examclass", data.exam=exam)
}

#Metodos
   set.Question<-function (object, i, valor){
        object@data.exam[i,1]<-valor
   }

   set.Answer<-function (object, i, j, valor){
     object@data.exam[i,j]<-valor
   }

   set.Score<-function (object, i, j, valor){
     object@data.exam[i,j]<-valor
   }

   set.ACorrect<-function (object, i, j, valor){
     object@data.exam[i,j]<-valor
   }

### Exam
```

GeneradorDeExamen

```
###GeneradorDeexams

Generate <- function(examen,no.questions,no.answers,no.exams) { ## SP, EN
 exams<-list()
 for(k in 1:no.exams){
  preg <- sample(1:no.questions, no.questions, replace=F)
  examAux<-exam( exam=matrix(c(""),nrow = no.questions, ncol=no.answers+4))
  for(i in 1:no.questions){
   correcta = -1
   examAux@data.exam[i,1]=get.Question(examen[[preg[[i]]]])
   res <- sample(1:(no.answers+1), (no.answers+1), replace=F)
   examAux@data.exam[i,2]=get.Answer(examen[[preg[[i]]]],res[[1]],1)
   if(res[[1]]==1)
    correcta=1
   for(j in 2:(no.answers+2)){
    examAux@data.exam[i,j]=get.Answer(examen[[preg[[i]]]],res[[(j-1)]],1)
    if(res[[j-1]]==1){
     correcta=j-1
    }
   }
   examAux@data.exam[i,no.answers+3]=get.Score(examen[[preg[[i]]]])
   examAux@data.exam[i,no.answers+4]=correcta
  }
  exams[[k]]=examAux
 }
 exams
}

###GeneradorDeExamenes
```

GeneradorDeDocumento

```
### GeneradorDeDocumento
## tex files for test

## library(Sweave)

## forzar que los ejercicios comienzan en pag impar
Generador <- function(examenes, anexos = "anexos.txt", skip = 0, quizname = "test",
  ANEXOS = FALSE, NEWPAGE = FALSE,
  no.examenes = 1, no.preguntas = 10, no.respuestas = 4,
  ASIGNAT = "Modelos de Colas,\\\\ Probabilidades y Estad\\'{\\i}stica II",
  DATE = "Mi\\'ercoles 05 de Diciembre 2012",
  TIMEQUIZ = 120,
  MERGE.SEED = 512,
  GROUP = "..................................",
  fontsize = 10,
  fontkey = "normalsize",
  topmargin = -1.0,
  textheight = 22.0,
  textwidth = 18,
  oddsidemargin = -1.0,
  baselinestretch = 1.0,
  cols = 2,
  MESS1 =  "Todas las preguntas valen igual. La pregunta cuya respuesta seleccionada
est\\'e mal resta {\\bf 1/3}. ",
  MESS2 = "Escoger la respuesta v\\'alida marcando claramente a la izquierda con {\\bf
X}:\\\\\\\\n",
  LANG="SP", TRZ = FALSE) { ## SP, EN
  testname = quizname
  for( num in 1:no.examenes ){
          nu.exam = num
          examen=examenes[[nu.exam]]
          quizname = paste(testname,nu.exam,sep="-")
          print("*************************************")
          print("*Comienza la generacion de documentos*")
          print("*************************************")
          ###
          sink( paste( quizname, ".txt", sep=""))
          cat("% random test.\n")
          cat(paste( "\\documentclass[", fontsize, "pt]{article}\n", sep=""))
          cat("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%\n")
          cat("\\usepackage[dvipdfm]{graphicx}\n")
          cat("\\usepackage[ansinew]{inputenc}\n")
          cat("\\usepackage[activeacute,spanish]{babel}\n")
```

```
        cat("\\usepackage{latexsym,multicol,amsmath,eurosym}\n")
        cat("\\usepackage{Sweave}\n")
        cat("\\decimalpoint \n")
        cat("\\usepackage{ifpdf}\n")
        cat("\\ifpdf%\n")
        cat("\\usepackage{pdflscape}\n")
        cat("\\else\n")
        cat("\\usepackage{lscape}\n")
        cat("\\fi\n")
        cat(paste( "\\topmargin=", topmargin, "cm \n", sep=""))
        cat(paste( "\\textheight=", textheight, "cm \n", sep=""))
        cat(paste( "\\textwidth=", textwidth, "cm \n", sep=""))
        cat(paste( "\\oddsidemargin=", oddsidemargin, "cm \n", sep=""))
        cat(paste( "\\renewcommand{\\baselinestretch}{", baselinestretch, "} \n",
sep=""))
        cat("\\begin{document}\n\n")
        ###
        cat("\\begin{center}\n")
        if ( LANG=="SP")
          cat("{\\Large Examen Versi\\'on:",nu.exam,"}\\\\\\\n")
        else
          cat("{\\Large Quiz Version",nu.exam,"}\\\\\\\n")
        cat("\\end{center}\n\n\n")
        if ( LANG=="SP")
          cat("{\\Large N\\'umero:..... Grupo:", GROUP,
"}\\\\\n")
        else
          cat("{\\Large Number:..... Group:", GROUP,
"}\\\\\n")

        if ( LANG=="SP")
          cat("{\\Large ... Apellidos:}\\\\\n")
        else
          cat("{\\Large ... Surname:}\\\\\n")
        if ( LANG=="SP")
          cat("{\\Large ... Nombre:}\\\\\n")
        else
          cat("{\\Large ... Name:}\\\\\n")
        cat("\\begin{center}\n")
        if ( LANG=="SP")
          cat("\\textbf{{\\Large ", ASIGNAT, " ( $ | n* $", nu.exam, ")}}\\\\\\\n", sep="")
## e: key-code
```

```
          else
            cat("\\textbf{{\\Large ", ASIGNAT, " ( $ | n* $", nu.exam, ")}}\\\\\\n", sep="")
## e: key-code
          if ( LANG=="SP")
             cat( DATE, ", Tiempo: ", TIMEQUIZ, " minutos \n")
          else
             cat( DATE, ", Time: ", TIMEQUIZ, " minutes \n", sep="")
          cat("\\end{center}\n\n\n")
          cat(paste( MESS1, sep=""))
          cat(paste( MESS2, "\\\\\\\\n", sep=""))
          cat("/////////////////////////////////////////////////////////////////////////////\n")
          cat(paste( "\\begin{multicols}{", cols, "}{\n", sep=""))
          cat("\\begin{enumerate}\n")
          for( p in 1:no.preguntas) {
            cat("\\item   ")
            cat( as.character( examen@data.exam[p,1])," ")
            cat( as.character( examen@data.exam[p,no.respuestas+2]),"\n")
            cat("\\begin{enumerate}\n")
            for( q in 1:no.respuestas) {
                    cat( "\\item  ")
                    cat( as.character( examen@data.exam[p,q+1]),"\n")
            }### q
            cat( "\n")
            cat("\\end{enumerate}\n")
          }### p
          cat("\\end{enumerate}\n")
          cat("}\n\\end{multicols}\n")
          cat("\\newpage\n")
          if (NEWPAGE) {
            cat(".")
            cat("\\newpage\n")
          }
          cat("\\end{document}\n")
          sink()
          print("Examen terminado.")

##################################################################
########  Solve  ################################################
       print("Comenzando la generación de la solución.")
          sink( paste( quizname, "-KEY.txt", sep=""))##--
          cat("% random test key.\n")
          cat(paste( "\\documentclass[", fontsize, "pt]{article}\n", sep=""))
          cat("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%\n")
```

```
        cat("\\usepackage[dvipdfm]{graphicx}\n")
        cat("\\usepackage[ansinew]{inputenc}\n")
        cat("\\usepackage[activeacute,spanish]{babel}\n")
        cat("\\usepackage{latexsym,multicol,amsmath,eurosym}\n")
        cat("\\decimalpoint \n")
        cat("\\usepackage{ifpdf}\n")
        cat("\\ifpdf%\n")
        cat("\\usepackage{pdflscape}\n")
        cat("\\else\n")
        cat("\\usepackage{lscape}\n")
        cat("\\fi\n")
        cat(paste( "\\topmargin=", topmargin, "cm \n", sep=""))
        cat(paste( "\\textheight=", textheight, "cm \n", sep=""))
        cat(paste( "\\textwidth=", textwidth, "cm \n", sep=""))
        cat(paste( "\\oddsidemargin=", oddsidemargin, "cm \n", sep=""))
        cat(paste( "\\renewcommand{\\baselinestretch}{", baselinestretch, "} \n",
sep=""))
        cat("\\begin{document}\n\n")
        cat("\\begin{center}\n")
        if ( LANG=="SP")
          cat("{\\Large Examen Versi\\'on:",nu.exam,"}\\\\\\\n")
        else
          cat("{\\Large Quiz Version",nu.exam,"}\\\\\\\n")
        cat("\\textbf{{\\Large ", ASIGNAT,"}}\\\\\\\n")
        if ( LANG=="SP")
          cat( DATE, ". Respuestas Correctas, Grupo: ", GROUP, ". Opci\\'on correcta
(a) \n")
        else
          cat( DATE, ". Correct Answers, Group: ", GROUP, ". Correct option (a) \n")
        cat("\\end{center}\n\n\n")
        cat("////////////////////////////////////////////////////////////////////////////////\n")
        if (ANEXOS)
          cat("\\input{", anexos, "}", sep="")## formularios, soluciones, comentarios,....
        cat(paste( "\\begin{multicols}{", cols, "}{\n", sep=""))
        cat("\\begin{enumerate}\n")
        for( p in 1:no.preguntas) {
          cat("\\item   ")
          cat( as.character( examen@data.exam[p,1])," (")
          cat( as.character( examen@data.exam[p,no.respuestas+2]),")\n", sep="")
          cat("\\begin{enumerate}\n")


          for( q in 1:no.respuestas) {
            cat( "\\item")
```

```
            if(q==examen@data.exam[p,no.respuestas+3])
                cat("(V) ", sep="")
            else
                cat("(X) ", sep="")
            cat( as.character( examen@data.exam[p,q+1]),"\n")
          }### q
          cat( "\n")
          cat("\\end{enumerate}\n")
        }
        cat("\\end{enumerate}\n")
        cat("}\n\\end{multicols}\n")
        cat("\\newpage\n")
        if (NEWPAGE) {
          cat(".")
          cat("\\newpage\n")
        }
        cat("\\end{document}\n")
        sink()
        print("Generado el examen con las respuestas.")
        ### post-proc
        print("****************************************")
        print("***************  Perl  ***************")
        print("****************************************")
        system( paste( "perl latexpand ", quizname, ".txt > ", quizname, ".Rnw", sep=""),
ignore.stdout = FALSE, ignore.stderr = FALSE) ## input{}
        system( paste( "perl latexpand ", quizname, "-KEY.txt", " > ", quizname, "-
KEY.Rnw", sep=""), ignore.stdout = FALSE, ignore.stderr = FALSE) ## input{}
        print("****************************************")
        print("**************  Sweave  **************")
        print("****************************************")
        system( paste( "R CMD Sweave --pdf ", quizname, ".Rnw", sep=""),
ignore.stdout = FALSE, ignore.stderr = FALSE) ## ---> tex
        system( paste( "R CMD Sweave --pdf ", quizname, "-KEY.Rnw", sep=""),
ignore.stdout = FALSE, ignore.stderr = FALSE) ## ---> tex
        print("****************************************")
        print("**********  latex y dvipdfm  **********")
        print("****************************************")
        ###
        system( paste( " latex      ", quizname, ".tex", sep=""), ignore.stdout = FALSE,
ignore.stderr = FALSE)
        system( paste( " dvipdfm -o ", quizname, ".pdf ",  quizname, ".dvi", sep=""),
ignore.stdout = FALSE, ignore.stderr = FALSE)
        ###
```

```
        system( paste( " latex      ", quizname, "-KEY.tex", sep=""), ignore.stdout =
FALSE, ignore.stderr = FALSE)
        system( paste( " dvipdfm -o ", quizname, "-KEY.pdf ", quizname, "-KEY.dvi",
sep=""), ignore.stdout = FALSE, ignore.stderr = FALSE)
        print("Limpiando archivos.")
        system( paste( " rm ",quizname, ".txt", sep=""), ignore.stdout = FALSE,
ignore.stderr = FALSE)
        system( paste( " rm ",quizname, ".tex", sep=""), ignore.stdout = FALSE,
ignore.stderr = FALSE)
        system( paste( " rm ",quizname, "-KEY.txt", sep=""), ignore.stdout = FALSE,
ignore.stderr = FALSE)
        system( paste( " rm ",quizname, "-KEY.tex", sep=""), ignore.stdout = FALSE,
ignore.stderr = FALSE)
        system( paste( " rm  *.aux ", sep=""), ignore.stdout = FALSE, ignore.stderr =
FALSE)
        system( paste( " rm  *.log ", sep=""), ignore.stdout = FALSE, ignore.stderr =
FALSE)
        system( paste( " rm  *.dvi ", sep=""), ignore.stdout = FALSE, ignore.stderr =
FALSE)
        system( paste( " rm  *.Rnw ", sep=""), ignore.stdout = FALSE, ignore.stderr =
FALSE)
  }
}

### GeneradorDeDocumento
```

2 Ayuda

Descripción de gentest

Information on package "gentest"

Description:

Package: gentest
Type: Package
Title: Test Generator
Version: 1.0
Date: 2013-12-22
License: GPL(>=2)
Author: Victor Navarro Ortego
Maintainer: Victor <v.n.ortego@gmail.com>
Description: Generate as many copies as you need a test and
solutions, you just need to have a xls file with
questions
License: What license is it under?
Built: R 2.14.1; ; 2014-01-03 19:36:54 UTC; unix

Index:

Conection	function to connect to a Database
DataDump	This function is used to get data from a workbook
Generador	Documents pdf generator, created with the exams.
Generate	Function that generates a list of exams
Mixer	Function that generates a test
MixerAnswers	Mixer answers Function
MixerQuestions	Mixer questions Function
exam	Constructor in class exam
get.Answer	Function that returns the answer
get.Question	Function that returns the question
get.Score	Function that returns the core
get.answerCorrect	Function that returns the correct answer
get.answersIncorrect	Function that returns the incorrect answer
get.data.question	Function that returns the question
get.data.score	Function that returns the score
get.questions	Function that returns the questions
question	Constructor of the class question
tableData	Constructor of the class tableData

?Conection

Conection package:gentest R Documentation

function to connect to a Database

Description:
This function is used to create a connection to an XLS file, so
that later can be modified or read that file.

Usage:
Conection(URI = "Banco.xls")

Arguments:
URI: URI is the String variable containing the path or filename XLS File

Value:
workbook: workbook variable type, which is used to access an XLS file

Note:
To use this method the XLConnect and rJava libraries are required. Function for the user's use.

Author(s):
Victor Navarro Ortego

Examples:

```
# con<-Conection("demoFiles/Banco.xls")
```

?DataDump

DataDump package:gentest R Documentation

This function is used to get data from a workbook

Description:
This function is used to get data from a workbook. EL worbook must be a XLS file with the following columns:
| NQuestion | Question | Correct | Answer1 | Answer2 | ... | Score | Topic |

Usage:
DataDump(file)

Arguments:
file: file is the workbook type variable, a variable must be connected to a valid xls

Value:
table: TableData type variable that contains the table with the data from the questions

Note:
Function for the user's use.

Author(s):
Victor Navarro Ortego

Examples:

```
# con<-Conection("demoFiles/Banco.xls")
# dat<-DataDump(con)
# print(dat@data)
```

?exam

exam package:gentest R Documentation

Constructor in class exam

Description:
Constructor in class exam, you need a variable of type matrix

Usage:
exam(exam)

Arguments:
exam: Type variable matrix, which contains the question, a list of the answers, the punctuation and the position of the correct answer

Value:
exam: A variable of type exam

Note:
Other functions of the exam class are: set.Question, set.Answer, set.Score, set.ACorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
exam(exam=matrix(c(""),nrow = 3, ncol = 4))

?Generador

Generador package:gentest R Documentation

Documents pdf generator, created with the exams.

Description:
This function receives a list of exams and some formatting options for generating pdf files with the versions of the tests ordered

Usage:
Generador(examenes, anexos = "", skip = 0, quizname = "", ANEXOS = FALSE, NEWPAGE = FALSE, no.examenes = 1, no.preguntas = 10, no.respuestas = 4, ASIGNAT = "", DATE = "2", TIMEQUIZ = 120, MERGE.SEED = 512, GROUP = ".", fontsize = 10, fontkey = "normalsize", topmargin = -1, textheight = 22, textwidth = 18, oddsidemargin = -1, baselinestretch = 1, cols = 2, MESS1 = "", MESS2 = "", LANG = "SP", TRZ = FALSE)

Arguments:
examenes: List elements of the exam class, each correspond to a different review
anexos: Name of the file containing the accompanying exam
quizname: Exam Name
ANEXOS: Boolean indication of the need for Annexere~~
no.examenes: Number of tests to generate
no.preguntas: Number of questions per exam
no.respuestas: Number of questions per question
ASIGNAT: Title exam
DATE: Date exam
TIMEQUIZ: Duration of exam

```
  cols: Number of columns exam
  MESS1: First paragraph of introduction
  MESS2: Second paragraph of introduction
  LANG: Language exam
  TRZ: Request a error traces

Value:
archives: Pdf files with exams and answers

Note:
    Function for the user's use.

Author(s):
    Victor Navarro Ortego

Examples:
    #   con<-Conection("demoFiles/Banco.xls")
    #   dat<-DataDump(con)
    #   examA<-Mixer(dat,4,5,3,4)
    #   examenes<-Generate(examA,4,3,3)
    #
    #   Generador( examenes,
    #     anexos = "refcard-io2012ColasMM.tex", skip = 0, quizname =
"rquizCMTC2013", ANEXOS = FALSE, NEWPAGE = FALSE,
    #     no.examenes = 2, no.preguntas = 4, no.respuestas = 4, ASIGNAT = "",
    #     DATE = "-------",      TIMEQUIZ = 50, MERGE.SEED = 512,
    #     GROUP = ".....", fontsize = 12, fontkey = "normalsize",
    #     topmargin = -1.75, textheight = 23.0, textwidth = 18, oddsidemargin = -1.0,
    #     baselinestretch = 1.0, cols = 2, MESS1 =  "", MESS2 = "",
    #     LANG="SP", TRZ = FALSE)
```

?Generate

Generate package:gentest R Documentation
Function that generates a list of exams

Description:
Function that generates a list of exams, and from mixed questions.

Usage:
Generate(examen, no.questions, no.answers, no.exams)

Arguments:
examen: List of objects question class, which represents the questions chosen for the test.
no.questions: Number of questions
no.answers: Number of answers
no.exams: Number of exams

Value:
exams: list containing objects of class exam

Note:
Function for the user's use.

Author(s):
Victor Navarro Ortego

Examples:
```
# con<-Conection("demoFiles/Banco.xls")
# dat<-DataDump(con)
# examA<-Mixer(dat,4,5,3,4)
# ex<-Generate(examA,4,3,3)
# print(ex[[2]]@data.exam)
```

?get.Answer

get.Answer package:gentest R Documentation

Function that returns the answer

Description:
Get function, which returns the answer of an object of the class question

Usage:
get.Answer(object, i, j)

Arguments:
object: Object of the class question
i: Number of row
j: Number of column

Value:
answer: String with the response requested

Note:
see also: question, get.Question, get.Score, get.Answer. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
questionaux<-question(quest="", answers=matrix(c(""),nrow = 2, ncol = 1), score="")
get.Answer(questionaux,1,1)

?get.answerCorrect

get.answerCorrect package:gentest R Documentation

Function that returns the correct answer

Description:
Get function, which returns the correct answer of an object of the class tableData

Usage:
get.answerCorrect(object, i)

Arguments:
object: Object of the class tableData
i: Number of row

Value:
answer: String with the correct answer

Note:
see also: tableData, get.data.question, get.questions, get.data.score, get.answersIncorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

See Also:
see also: tableData get.data.question get.questions get.data.score get.answersIncorrect

Examples:
con<-Conection("demoFiles/Banco.xls")
dat<-DataDump(con)
get.answerCorrect(dat,1)

?get.nswersIncorrect

get.answersIncorrect package:gentest R Documentation

Function that returns the incorrect answer

Description:
Get function, which returns the List with incorrect answer of an object of the class tableData

Usage:
get.answersIncorrect(object, i, column)

Arguments:
object: Object of the class tableData
i: Number of row
column: Number of column

Value:
answer: List with the incorrect answer

Note:
see also: tableData, get.data.question, get.questions, get.data.score, get.answersCorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
con<-Conection("demoFiles/Banco.xls")
dat<-DataDump(con)
get.answerIncorrect(dat,1,1)

?get.data.question

get.data.question package:gentest R Documentation

Function that returns the question

Description:
Get function, which returns the question of an object of the class tableData

Usage:
get.data.question(object, i)

Arguments:
object: Object of the class tableData
i: Number of row

Value:
answer: String with the question

Note:
see also: tableData, get.questions, get.data.score, get.answersCorrect, get.answersIncorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
con<-Conection("demoFiles/Banco.xls")
dat<-DataDump(con)
get.data.question(dat,1)

?get.data.score

get.data.score package:gentest R Documentation

Function that returns the score

Description:
Get function, which returns the score of an object of the class tableData

Usage:
get.data.score(object, i)

Arguments:
object: Object of the class tableData
i: Number of row

Value:
score: String with the score of question

Note:
see also: tableData, get.questions, get.data.score, get.answersCorrect, get.answersIncorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
```
# con<-Conection("demoFiles/Banco.xls")
# dat<-DataDump(con)
# get.data.score(dat,1)
```

?get.Question

get.Question package:gentest R Documentation

Function that returns the question

Description:
Get function, which returns the question of an object of the class question

Usage:
get.Question(object)

Arguments:
object: Object of the class question

Value:
question: String with the question

Note:
see also: question, get.Score, get.Answer. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
```
questionaux<-question(quest="", answers=matrix(c(""),nrow = 2, ncol = 1), score="")
get.Question(questionaux)
```

?get.questions

get.questions package:gentest R Documentation

Function that returns the questions

Description:
Get function, which returns the question list of an object of the class tableData

Usage:
get.questions(object, topic)

Arguments:
object: Object of the class tableData
topic: topic that the answers are ordered

Value:
questions: Questions list

Note:
see also: tableData, get.data.question, get.data.score, get.answersCorrect, get.answersIncorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
```
# con<-Conection("demoFiles/Banco.xls")
# dat<-DataDump(con)
# get.questions(dat,"")
```

?get.Score

get.Score package:gentest R Documentation

Function that returns the core

Description:
Get function, which returns the score of an object of the class question

Usage:
get.Score(object)

Arguments:
object: Object of the class question

Value:
score: String with the score

Note:
see also: question, get.Question, get.Answer. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
```
questionaux<-question(quest="", answers=matrix(c(""),nrow = 2, ncol = 1), score="")
get.Score(questionaux)
```

?Mixer

Mixer package:gentest R Documentation

Function that generates a test

Description:
Given the number of questions, the number of responses and the subject, in an array with questions and answers in an examination.

Usage:
Mixer(datas, no.questions, no.max.questions, no.answers, no.max.answers, topic = "")

Arguments:
datas: number of questions
no.questions: number of questions
no.max.questions: maximum number of questions
no.answers: number of answers
no.max.answers: maximum number of answers
topic: topic of exam questions

Value:
exam: Matrix with the questions, answers and scores

Note:
see also: MixerQuestions MixerAnswers. Function for the user's use.

Author(s):
Victor Navarro Ortego

Examples:
con<-Conection("demoFiles/Banco.xls")
dat<-DataDump(con)
Mixer(dat,4,5,3,4)

?MixerAnswers

MixerAnswers package:gentest R Documentation

Mixer answers Function

Description:
Indicated a question questions a list

Usage:
MixerAnswers(datas, no.question, no.max.answers, no.answers)

Arguments:
datas: number of questions
no.question: number of question
no.answers: number of answers
no.max.answers: maximum number of answers

Value:
answers: List of question responses indicated mixed

Note:
see also: Mixer, MixerQuestions. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
con<-Conection("demoFiles/Banco.xls")
dat<-DataDump(con)
MixerAnswers(dat,1,4,3)

?MixerQuestion

MixerQuestions package:gentest R Documentation

Mixer questions Function

Description:
Indicated a question returns a list of mixed responses

Usage:
MixerQuestions(datas, no.questions, no.max.questions, topic = "")

Arguments:
datas: number of questions
no.questions: number of questions
no.max.questions: maximum number of questions
topic: topic of exam questions

Value:
questions: List of questions mixed

Note:
see also: Mixer, MixerAnswers. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
con<-Conection("demoFiles/Banco.xls")
dat<-DataDump(con)
MixerQuestions(dat,2,3)

?question

question package:gentest R Documentation

Constructor of the class question

Description:
Constructor of the class question

Usage:
question(quest, answers, score)

Arguments:
quest: String with question
answers: List with answers
score: String with score

Value:
question: Object of class question

Note:
see also: get.Question, get.Score, get.Answer. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
questionaux<-question(quest="", answers=matrix(c(""),nrow = 2, ncol = 1), score="")

?tabledata

tableData package:gentest R Documentation

Constructor of the class tableData

Description:
Constructor of the class tableData

Usage:
tableData(data)

Arguments:
data: Object of class dataframe

Value:
table: TableData object type

Note:
see also: get.data.question, get.questions, get.data.score, get.answersCorrect, get.answersIncorrect. Internal function of package.

Author(s):
Victor Navarro Ortego

Examples:
file<-Connection("demoFiles/Banco.xls")
tableData(readWorksheet(file, "banco"))

BIBLIOGRAFÍA

[1] IEEE-STD-830-1998 : ESPECIFICACIONES DE LOS REQUISITOS DEL SOFTWARE

[2] John M. Chambers "Software for Data Analysis Programming with R"

[3] Leslie Lamport,. "The Writings of Leslie Lamport: LaTeX: A Document Preparation System". Avible: Leslie Lamport's Home Page. Retrieved 2007-04-27.

[4] R Development Core Team (2011). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, Available: http://www.R-project.org/.

[5] Excel, Available: http://office.microsoft.com/en-001/excel/

[6] GenerTest, Available: https://sites.google.com/site/lara3107/Home/software/genertest

[7] LaTex, Available: http://www.latex-project.org/

[8] MikTex, Available: http://miktex.org/

[9] Moodle, Avaible: http://docs.moodle.org/25/en/Test

[10] WebQuestion 2.0, Avaible: http://www.aula21.net/webquestions/

[11] Wikipedia, Excel, Available: http://en.wikipedia.org/wiki/Microsoft_Excel

[12] Wikipedia, LaTeX, Available: http://en.wikipedia.org/wiki/LaTeX

[13] Wikipedia, Perl, Available: http://en.wikipedia.org/wiki/Perl

[14] XLConnect, Available: http://www.mirai-solutions.com/site/index.cfm/id_art/60255

[15] Generación automática de reportes con R y LATEX Automatic report generation with R and LATEX Mario Alfonso Morales Rivera.

[16] Chambers JM, “How S4 methods work.” available on CRAN, 2006.

[17] "The R User Conference, useR!" , Spain Book of Contributed Abstracts, University of Castilla-La Mancha, Albacete, July 10-12 2013

[18] Manual para generación de paquetes en R, Available: http://cran.us.r-project.org/doc/manuals/R-exts.html

[19] Robin K. S. Hankin. A step-by-step guide to writing a simple package that uses S4 methods: a \hello world" example, URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.190.6655

[20] Thomas Baier and Erich Neuwirth. Excel :: Com :: R. Computational Statistics, 2007, ch. 22, pp. 91–108.

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
	Fecha/Hora	Wed Jan 08 16:59:10 CET 2014
	Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
	Numero de Serie	630
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)